# Distributed Network Management Security

Paul Meyer

Secure Computing Corporation
Telephone: (612) 628-2700
Fax: (612) 628-2701
e-mail: Paul_Meyer@securecomputing.com

Contracting U.S. Government Agency: U.S. Army Missile Command
Funding U.S. Government Agency: DARPA
Contract Number: DAAH01-95-C-R196
Approved for Public Release - Distribution Unlimited

**Abstract**

Use of SNMP to securely manage distributed networks through firewalls has not been formally described, although features critical to such management are included in SNMP. This document reports on a study performed at Secure Computing Corporation on a method to solve this management function. The project name this study occurred under is Distributed Network Management Security.

Slight modifications to the SNMP V2 User-Based Security Model (RFC 1910) and a conceptual redeployment of some of the functions contained within this model provide a basis for this study. The acronym DNMS will be used in this document to refer to the modifications.

The basis for the DNMS extensions is a firewall platform that contains at least two distinct network stack implementations, one for the exterior, or public network, and one for the interior, or protected network. DNMS consists of two SNMP V2 proxies, one on each network, with the security-related functions implemented in a third component that also serves as the communication path between the two proxy components.

This implementation allows the management and use of SNMP security to be concentrated in the firewalls, where it is assumed that the threats being protected against lie outside the firewall.

# Contents



# List of Figures

# 1 Overview

The goal of Distributed Network Management Security (DNMS) is to aid in managing and securing networks. It relies on existing firewall technology and network management solutions with some modifications.

Most corporations today are attempting to secure and manage networks with a number of geographically distributed locations. At the same time, they are relying more and more on the Internet as a means of reducing their local network costs, both in terms of facilities and administration.

Traditional means of securing these distributed sites involve the use of firewalls. These firewalls are not able to secure the network management traffic, primarily because the state of the base network management protocol (SNMP) with respect to security has been problematic.

The result is that a number of strategies are used. Most often, local management of the sites is used with manual coordination between them (racking up phone bills as the network administrators consult with each other). Often, "backdoors" to the firewall are used, such as dedicated links between sites or dial-in modems. Occasionally the firewall is instructed to let **all** SNMP traffic in, essentially defeating any reason for having a firewall in the first place.

The DNMS program is currently refining the concepts of the program by building a prototype implementation of the system. An earlier phase generated a White Paper[1] [2] that outlined the basic strategy used in DNMS. Figure 1 shows this solution. DNMS operates on the two intermediate firewalls, securing the networm management traffic between the management station and the distributed agents.

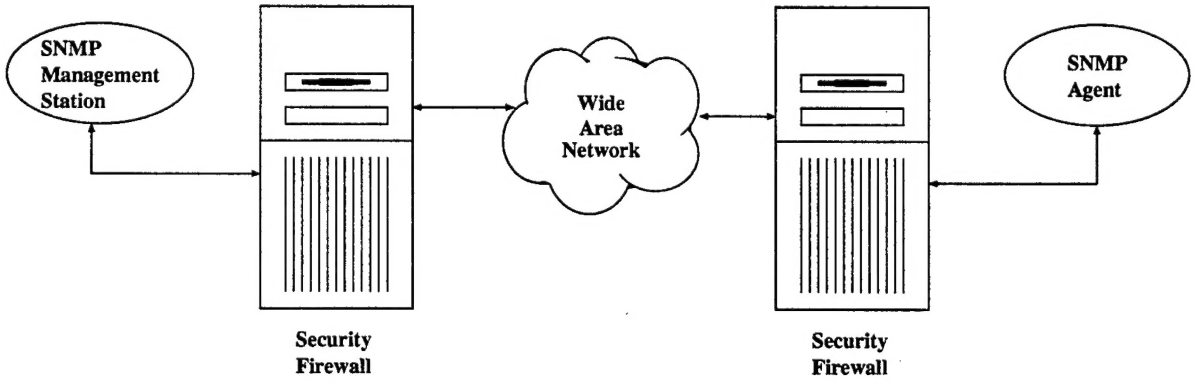


Figure 1: DNMS in a Network

DNMS is intended to facilitate remote management of systems when both the Network Management Station (NMS) and the managed device are protected by firewalls from the external network. DNMS does not imply that the NMS must be protected by a firewall. A requirement of the DNMS design is that DNMS functionality must be applicable to the NMS, allowing it to exist anywhere in the network. Any SNMP protocol modifications required by DNMS must therefore be brought forward to the Internet community for consideration as inclusion to the standard.

Implementation for DNMS will meet the following objectives:

- Demonstrate the feasibility of the DNMS concept by building a demonstration prototype showing secure management through firewalls.

[1] http://www.securecomputing.com/dnms/final-report.html

- Provide the solution in such a manner that the network management system and the managed devices are unaware of the firewall.
- Show that DNMS will interoperate with other network components using the Simple Network Management Protocol (SNMP)[9][7].
- Publicize the results of this effort to the Internet community and solicit its feedback prior to productization of the technology.
- Demonstrate that network management can successfully be implemented on a firewall whose access is protected by Type Enforcement.
- Provide a measure of security for managed devices without requiring that these devices upgrade to secure versions of SNMP.

The solution describes DNMS in terms of SNMP proxies. Elements of DNMS are:

- A Network Proxy. This element is responsible for ensuring that SNMP messages for SNMP entities behind the firewall are correctly delivered to that entity when the messages are actually sent to the firewall.
- An Administration Proxy. This element verifies access control to SNMP entities, such as the SNMP "view" of that entity.
- A Cryptographic Service Proxy. This element handles the authentication, integrity, and confidentiality requirements of DNMS.

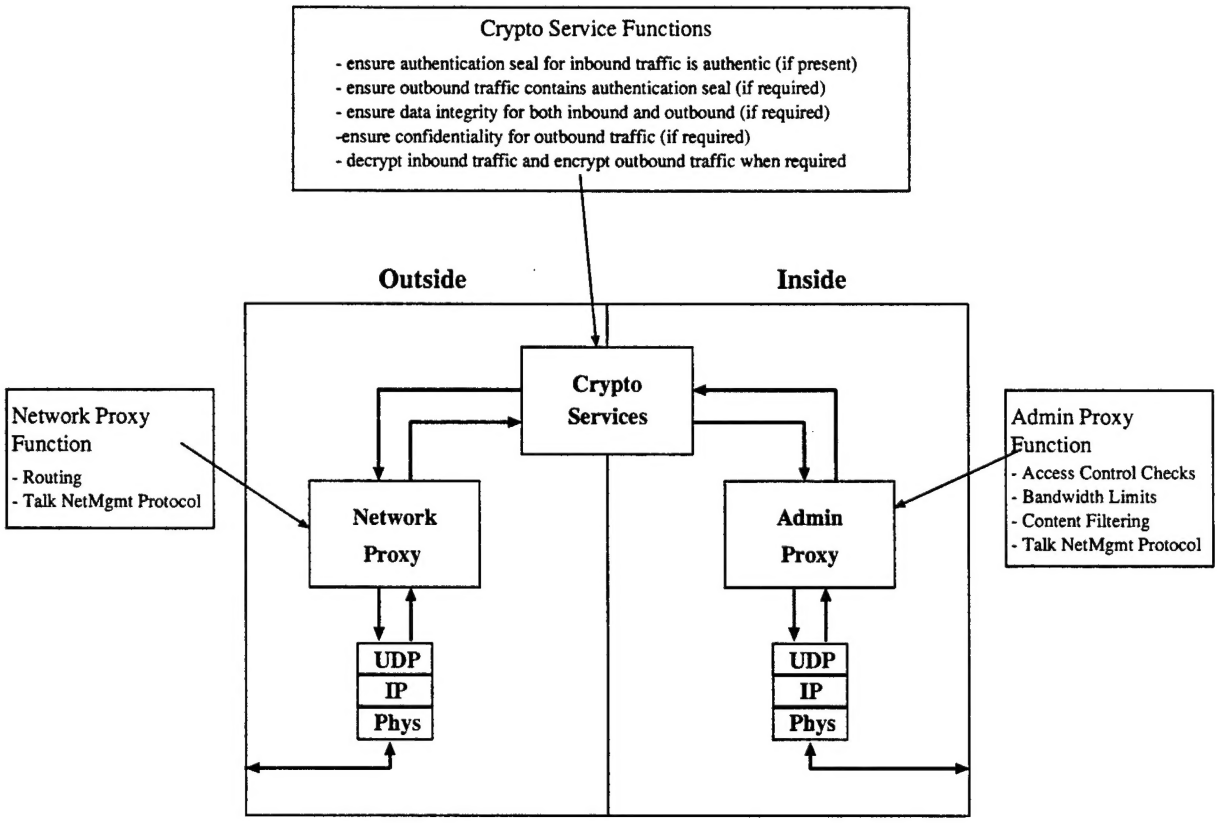


Figure 2: DNMS Functional Architecture

Figure 2 shows this functional decomposition of DNMS.

A number of other site policy controls can be implemented in DNMS. These could include filtering of management information, management traffic utilization control, time-restricted accesses, etc.

The end result is that all network management traffic traversing the Wide Area Network between the distributed sites will, at site policy discretion, be encrypted. Management stations, and more importantly, agents, need not be upgraded to versions of SNMP that handle authentication and encryption on an individual entity basis.

# 2 Prototype Development Activities

## 2.1 Analysis and Requirements

This phase of the DNMS program started with an analysis of earlier documentation along with the state of SNMP as of January 1996. Both of the contending SNMP V2 security proposals, known as SNMPv2usec and SNMPv2*, were examined for suitability with DNMS. These stand for "User-based Security Model for SNMPv2" (RFCs 1909 and 1910)[8] and "A Synthesis Security Model" (series of IETF Internet Drafts[5]) respectively. This analysis was particularly important as earlier work on DNMS was closely linked to the historic version of SNMP from RFCs 1441 through 1452.[1]. It appeared that either of the proposals could easily be modified to carry sufficient information to obtain user information from some key distribution mechanism (such as X.509 certificates).

The result of this analysis was a Requirements Document[2][4]. Here are some of the key requirements from this document.

- DNMS will be based upon the SNMP V2 protocol as defined in RFCs 1901 - 1908.
- DNMS will utilize a flexible administration model on top of the SNMP V2 protocol operations.
- DNMS will utilize security extensions that support public key cryptography.
- DNMS will support an internal (SNMP-based) method for certificate maintenance.
- DNMS will support an external (non-SNMP) protocol for certificate maintenance.
- DNMS will support (via proxy) agents that communicate with the SNMP V1 protocol.
- DNMS will support (via proxy) agents that communicate with the SNMP V2 protocol using the community-based administration and authentication model[6].
- DNMS will validate the integrity of messages received on the external network according to the security policy in force.
- DNMS will authenticate messages received on the external network according to the security policy in force.
- DNMS will forward messages though the firewall that pass the integrity and authentication checks.
- DNMS will provide encryption and decryption proxy services for systems behind the firewall.
- DNMS will supply a local cache for certificates if required by the local security policy.

## 2.2 Design

SCC contacted a number of vendors of SNMP agent code to try to reduce the time spent in base agent implementation in order to develop the DNMS prototype. Mostly due to the uncertainty of the SNMP market, commercial implementations were not readily available.

Protocol change design was quite easy. However, the logical design of two SNMP-aware proxies and the distinct cryptographic processing subsystem was not fitting into the modular design of SNMPv2*. Without a working implementation to examine, we decided to concentrate on the SNMPv2u

[2] http://www.securecomputing.com/dnms/reqs.ps

model. A very limited implementation of the model was available. This was the reference SNMPv1 implementation provided from Carnegie Mellon University (CMU) that had the SNMPv2 protocol modifications made but only portions of SNMPv2usec. The DNMS team decided to gain some experience by working with the CMU implementation. Vendor contacts were still being attempted during this time.

Details for these protocol modifications are found in section 3. A Technical Report[3][3] was also written to detail the planned modifications.

## 2.3 Prototype Implementation

Due to the difficulty in getting a commercial implementation to work with, SCC continued to work with the CMU distribution.

The cryptographic process subsystem proved to be fairly easy to implement. Cryptographic processing was placed within a generic proxy skeleton and unit tested successfully. This initial version did no caching and no Directory lookups, but locations were identified where such calls could easily be inserted in the code. Certificate caching and an X.500 DUA both exist within SCC's technology; integrating these is of high priority in continuing work on DNMS.

The use of the CMU code became more problematic. The DNMS modifications required SNMPv2 context checking be added to the code at the same time the interfaces to the crypto proxy were being implemented. Configuration of the two instances of the CMU code (one for the network proxy and the other for the admin proxy) were combined to make the initial setup of DNMS easier to understand.

The target platform for the DNMS prototype is a Secure Computing Sidewinder firewall, which utilizes Type Enforcement. Each of the proxies runs in a distinct domain. The network and admin proxy domains have extremely limited access; essentially they have network access only to their attached network stack and read access to their configuration files. The crypto proxy requires slightly more access to communicate with the other proxies, the Directory, and the certificate cache.

Section 4 describes the operations necessary to pass messages through a DNMS system.

Existing funding for DNMS ran out just as messages were starting to be successfully transmitted through the initial two firewall setup. Despite this, a number of things were learned in the prototype. These will be detailed in section 5.

[3] http://www.securecomputing.com/dnms/sci-tech.ps

# 3 SNMP Protocol Modifications

The primary modification to the SNMPv2usec header is to provide semantics on the userName field. This modification would allow the use of constructs such as the X.500 Distinguished Name in the userName, which in turn would allow the use of a certificate infrastructure as an adjunct to the SNMPv2usec model for key distribution. The form of the userName was taken from a draft of the PKIX working group.

SNMPv2usec defines the header in terms of an OCTET STRING rather than using the SNMPv2 SMI. A direct copy of the GeneralName ASN.1 construct was not used due to the confusion a standard BER-encoded field could cause. Instead, the field was defined as follows:

```
<userLen>  a one octet value containing the length of the <userName>
<userType> an octet value indicating which of the GeneralName
           choices the <userName> consists of.
<userName> the string representation of the <userName>.  X.500
           Distinguished Names and X.400 O/R Addresses are shown
           as defined in RFC 1779 and RFC 1685.
```

DNMS chose a <model> value of 99 to indicate the different header.

The GeneralName construct is defined as the following:[4]

```
GeneralName  ::=  CHOICE  {
     otherName        [0] INSTANCE OF OTHER-NAME,
     rfc822Name       [1] IA5String,
     dNSName          [2] IA5String,
     x400Address      [3] ORAddress,
     directoryName    [4] Name,
     ediPartyName     [5] IA5String,
     url              [6] IA5String }
```

Use of GeneralName choice 0 (otherName) indicates use of a standard SNMPv2usec <userName>.

The final modification is to potentially allow either SNMPv2 PDUs or SNMPv1 PDUs to be contained inside the protected data field. This was intended for proxy applications. Rules for translating between SNMPv1 and SNMPv2 are occasionally ambiguous and other people are looking at this particular problem.

[4] The PKIX working group will be changing the definition of this construct; DNMS will be updated as this occurs.

The full DNMS header is shown below.

```
Message ::=
     SEQUENCE{
          version
               INTEGER { v2 {2} },
          parameters
               OCTET STRING,
          -- <model=99>
          --      <qoS><agentID><agentBoots><agentTime><maxSize>
          --      <userLen><userType><userName><authLen><authDigest>
          --      <contextSelector>
          data
               CHOICE {
                       plaintext
                            PDUs,
                       encrypted
                            OCTET STRING
                       }
       }
```

# 4 DNMS Implementation Details

Figure 3 is shown here to aid in following the discussion.

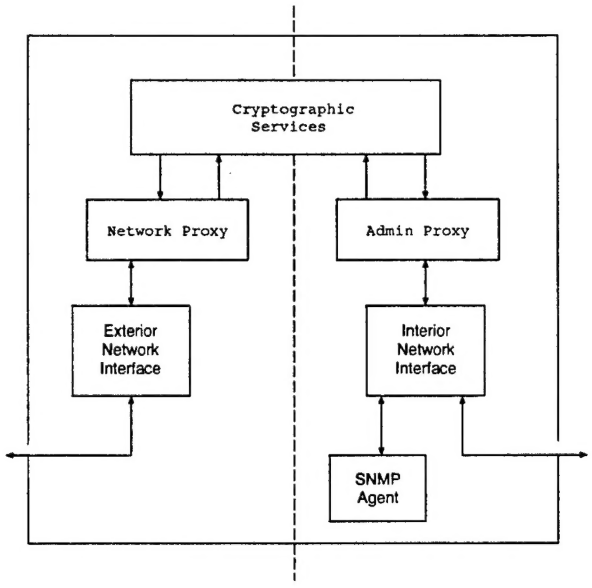


Figure 3: DNMS Implementation

## 4.1 Messages Passing Interior to Exterior

Messages passing from the interior network to the exterior network need to be transformed from their base SNMP format to DNMS, with the desired cryptographic processing applied to the messages.

The admin proxy will accept the message from the transport layer as usual and begin processing according to SNMP rules.

Received SNMPv2usec messages are minimally transformed. The Local Configuration Datastore (LCD) must contain information for the user. If none exists (i.e. no secrets for this user), the message cannot be transported if authentication or encryption are required. In the LCD, find the matching localContext entry, and get the corresponding remoteContext. No match indicates an error. The configuration of the firewall should allow no local type contexts to the admin proxy; they should have been passed to the SNMP agent. Rebuild the SNMPv2usec header into a DNMS header. The **<userName>** is set to **otherName**. **<agentID>** is set to the receiving DNMS, **<agentBoots>** and **<agentTime>** are unmodified.

SNMPv2c and SNMPv1 messages have similar transforms applied to them. A preconfigured mapping between the community name and the DNMS remoteContext must exist. When this mapping is found in the LCD, pull up the matching remoteContext, **<qoS>** and **<userName>**, along with the last indicators for **<agentBoots>** and **<agentTime>** (these are required for maintaining protection against replay attacks). Use the values to build a DNMS header.

The message is then passed to the crypto proxy for authentication.

The crypto proxy first sets up an additional replay protection mechanism above and beyond **<agentTime>** handling in the admin proxy. Request type messages cause a time window to be

calculated within which report/response type PDUs are expected. Report/response type PDUs need to be received within this window to be transferred through the crypto proxy.

If no authentication or encryption is required on the DNMS message, it is passed on to the network proxy without modification.

When the message requires authentication, the **<userName>** is used to query for keying material. The crypto proxy will first search the LCD for a match, then will check a local certificate cache. Only when both these have failed will the crypto proxy look to an exterior certificate infrastructure (such as an X.500 directory) for keying material.

Authentication is then performed as defined in RFC1910 to build the **<authDigest>**.

If confidentiality protection is also required, the crypto proxy will obtain the privacy keying material. To avoid padding, confidentiality is provided by using DES in Cipher Feedback (CFB) mode. (If this proves too slow, a padding scheme will be implemented and either Electronic Codebook (ECB) or Cipher Block Chaining (CBC) mode will be used.) The data portion is then encrypted.

The resultant message is passed on to the network proxy. Based upon the **<contextSelector>** in the message, the network proxy will select the network address of the receiving DNMS. The message is then sent via the exterior network stack.

(We have used the convention of building the **<contextSelector>** out of the IP address that handles the context, a unique tag, and a corresponding value that would map to a view name. This reduces the number of configurable items - always a good thing in experimental work.)

## 4.2 Messages Passing Exterior to Interior

Messages passing from the exterior network to the interior network must be cryptographically correct. They then can be transformed from the DNMS format back to the original SNMP format for transmission on to the end management entity.

The network proxy will accept SNMPv1, SNMPv2c, and SNMPv2usec messages with a minimal agent that supports the system group only. This should give an indication to a management station that the system needs to communicate with DNMS protocol extensions to validly forward messages through the proxy.

DNMS messages should contain a local-proxy **<contextSelector>**. This **<contextSelector>**, along with the **<userName>**, should be known to the network proxy (via the LCD) in order to be accepted and passed to the crypto proxy.

The crypto proxy will pass messages passed through to the admin proxy if the **<qoS>** indicates that no confidentiality or authentication was applied to the message. As with outbound crypto processing, the message will be checked to verify it is within the time window before being passed through.

As with the outbound proxy, the **<userName>** is used to find keying materials in the same order (LCD, local cache, remote methods). The message is then decrypted (if necessary) and the <authDigest> validated. Messages successfully passing this processing are passed on to the admin proxy.

The admin proxy must use the passed <contextSelector> to find the transform in the LCD. For a DNMS to SNMPv2usec message, pick up the matching localContext, downgrade the **<qoS>** as configured, and rebuild the header to SNMPv2usec format (the **<agentID>** for the end entity is also in the LCD). Send the message. For a SNMPv2c or SNMPv1 message, pick the matching community string for the received **<contextSelector>**. Save the latest indications for **<agentBoots>** and **<agentTime>** for use on the next response or request. Rebuild the header in community format and send the message to the end entity. Unknown contexts at this point are silently discarded.

An existing SNMP agent on the firewall is also shown in Figure 3. This agent is treated identically to other management entities lying within the interior network.

## 4.3 Context Checking in CMU Distribution

Some simple changes were added to the CMU distribution to even provide for context checking. Both the network and admin proxies make use of these changes.

- Add context definitions to the **snmpd.conf** file. A context includes the type and the associated MIB view (what things it can see on the system). For local-proxy types that translate to SNMPv2usec, the configuration should indicate what the remote <snmpID> and remote <contextSelector> should be. Local-proxy translations for SNMPv1 and SNMPv2c need to indicate the remote IP address and community string that should be used.
- Future support should include this in a MIB that AUGMENTS the SNMPv2usec Remote Configuration MIB's usecContext Table.
- Verify <contextSelector> during the **check_auth()** processing.

## 4.4 Proxy Interfaces

This is a simple interface with the crypto proxy opening two sockets and the other two proxies communicating with the crypto proxy via these sockets. Sockets are objects that can be labelled via Type Enforcement, so only the correct processes can create (open), read, write, or destroy using the socket. This provides some level of assurance that only messages that pass the crypto proxy checks can be sent through the firewall.

# 5 Observations

The primary problem with managing SNMP security has been scale of deployment. The historic party model of SNMP did work, but only within a limited range.

As stated, one of the key points in DNMS was to reduce the key management areas of SNMP to utilize, at local option, some other method like an X.500 Directory and X.509 certificates[5]. It appears as if the context management has the potential of being equally onerous on the personnel managing the distributed network. The IETF's Distributed Management Working Group has issued a trial architecture document that would, if implemented, solve this area. The solution consists (in DNMS terms) of the admin proxy utilizing an autodiscovery protocol to find the management entities interior to the firewall, and using Informs to interact with a controlling management station and peer implementations of DNMS to build the context knowledge for the network.

This context knowledge for the network should be represented in a DNMS-specific MIB. Additional experience working with the prototype should generate more information about what else should be contained in this MIB.

One of the limiting factors in the current prototype implementation is in not handling SNMP Traps. There is considerable difference in the format of Traps between SNMPv1 and SNMPv2; proper support may mean adding translation capabilities between SNMPv1 and SNMPv2 to the admin proxy. Recent publication of RFC 2089[10] should generate some implementation experience DNMS can utilize for adding Trap support.

A filtering capability could be aligned with the cryptographic proxy subsystem. This feature would not allow user-definable 'critical' data to transit the firewall without the appropriate level of cryptographic services applied to the PDU carrying the data. Such a feature is important as MIBs can be defined to provide almost any information.

Use of IPSEC as an option for encryption of DNMS traffic should be examined. This could alleviate the possibility of traffic analysis against messages passing between DNMS systems. With the current SNMP-based encryption, portions of the message header are visible. DNMS would still be needed to provide the application-level authentication upon management operations.

[5] Some people believe moving key management around is a shell game; someone else can solve it! The idea in using X.500 Directories is that these appear closest to real deployment than other means, but want to provide flexibility for other solutions that have some potential.

# References


[1] J. Case, K. McCloghrie, M. Rose, and S. Waldbusser. Introduction to version 2 of the internet-standard network management framework. Technical Report RFC 1441, SNMP Research Inc., Hughes LAN Systems, Dover Beach Consulting Inc., Carnegie Mellon University, May 1993. Online information is available at ds.internic.net[6].

[2] Secure Computing Corporation. Distributed network management security white paper. Final Status Report CDRL: A002, Secure Computing Corporation, 2675 Long Lake Road, Roseville, Minnesota 55113-2536, October 1994.

[3] Secure Computing Corporation. Distributed network management security semi annual scientific and technical report. Final Submission CDRL: A003, Secure Computing Corporation, 2675 Long Lake Road, Roseville, Minnesota 55113-2536, May 1996.

[4] Secure Computing Corporation. Distributed network management security system requirements. Final Submission CDRL: N/A, Secure Computing Corporation, 2675 Long Lake Road, Roseville, Minnesota 55113-2536, December 1996.

[5] SNMPv2star Working Group. Administrative model for version 2 of the simple network management protocol (snmpv2). Technical report, IETF, December 1995. Online information is available at ietf.cnri.reston.va.us[7].

[6] J.Case, K. McClogrie, M. Rose, and S. Waldbusser. Introduction to community-based snmpv2. Technical Report RFC 1901, SNMPv2 Working Group, January 1996. Online information is available at ds.internic.net[8].

[7] J.Case, K. McClogrie, M. Rose, and S. Waldbusser. Protocol operations for version 2 of the simple network management protocol (snmpv2). Technical Report RFC 1905, SNMPv2 Working Group, January 1996. Online information is available at ds.internic.net[9].

[8] K. McClogrie. An administrative infrastructure for snmpv2. Technical Report RFC 1909, Cisco Systems, Inc., February 1996. Online information is available at ds.internic.net[10].

[9] M. Schoffstall, M. Fedor, J. Davin, and J. Case. A simple network management protocol (snmp). Technical Report RFC 1157, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, SNMP Research, May 1990. Online information is available at ds.internic.net[11].

[10] B. Wijnen and D. Levi. Mapping snmpv2 onto snmpv1 within a bi-lingual snmp agent. Technical Report RFC 2089, Network Working Group, January 1997. Online information is available at ds.internic.net[12].


[6] http://ds.internic.net/rfc/rfc1441.txt
[7] ftp://ietf.cnri.reston.va.us/internet-drafts/draft-various-snmpv2-adminv2-syn-01.txt
[8] http://ds.internic.net/rfc/rfc1901.txt
[9] http://ds.internic.net/rfc/rfc1905.txt
[10] http://ds.internic.net/rfc/rfc1909.txt
[11] http://ds.internic.net/rfc/rfc1157.txt
[12] http://ds.internic.net/rfc/rfc2089.txt